# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **5/26/24** | **Latoria Judkins** |  |

## Client



## Instructions

Submit this completed vulnerability assessment report. Replace the bracketed text with the relevant information. In this report, identify your security vulnerability findings and recommend the next steps to remedy the issues you have found.

* Respond to the five steps outlined below and include your findings.
* Respond using your own words. You may also include images or supporting materials. If you include them, make certain to insert them in the relevant locations in the document.
* Refer to the Project One Guidelines and Rubric for more detailed instructions about each section of the template.

## Developer

Latoria Judkins

**1. Interpreting Client Needs**

Determine your client’s needs and potential threats and attacks associated with the company’s application and software security requirements. Consider the following questions regarding how companies protect against external threats based on the scenario information:

* **What is the value of secure communications to the company?**

Secure communications are of high value or top priority with Artemis Financial since the company’s mission is “Security is everyone’s responsibility” and it will help with ensuring the customers that their sensitive information remains confidential.

* **Are there any international transactions that the company produces?**

Yes, there will be international transactions that the company produces since the software being used is for entrepreneurs, businesses, and government agencies around the world.

* **Are there governmental restrictions on secure communications to consider?**

Artemis Financial must follow any government regulations and restrictions when it comes to secure communications since they are a financial institution.

* **What external threats might be present now and in the immediate future?**

External threats that might be present now and in the immediate future are someone trying to gain unauthorized access to sensitive information from the customers and the company.

* **What modernization requirements must be considered, such as the role of open-source libraries and evolving web application technologies?**

The modernization requirements must be considered security, flexibility, scalability, and being able to integrate with existing software. The role of open-source libraries and evolving web application technology can be detrimental because a lot of the security threats or risk come from links or email that we think is safe.

**2. Areas of Security**

Refer to the vulnerability assessment process flow diagram. Identify which areas of security apply to Artemis Financial’s software application. Justify your reasoning for why each area is relevant to the software application.

**API**- allows software applications to communicate and exchange information with other applications. Artemis Financial uses RESTful to communicate with other software.

**Input Validation**- prevents attacks. Artemis Financial uses input validation to validate the user information.

**Code Error**- shows what needs to be fixed in API. Artemis Financial can use to reduce security risks in the software.

**3. Manual Review**

Continue working through the vulnerability assessment process flow diagram. Identify all vulnerabilities in the code base by manually inspecting the code.

* Not verifying user’s account which means there is weak authentication
* Not using HTTPS
* Not using cryptography
* Insecure data transmission

**4. Static Testing**

Run a dependency check on Artemis Financial’s software application to identify all security vulnerabilities in the code. Record the output from the dependency-check report. Include the following items:

* The names or vulnerability codes of the known vulnerabilities
* A brief description and recommended solutions provided by the dependency-check report
* Any attribution that documents how this vulnerability has been identified or documented previously

bcprov-jdk15on-1.46.jar-

* CVE-2024-34447
* CVE-2016-1000338
* CVE-2016-1000342
* CVE-2016-1000343
* CVE-2024-29857
* CVE-2016-1000344
* CVE-2016-1000352
* CVE-2024-30171
* CVE-2016-1000341
* CVE-2016-1000345
* CVE-2016-1000345
* CVE-2020-15522
* CVE-2020-0187
* CVE-2023-33202
* CVE-2020-26939
* CVE-2023-33201
* CVE-2016-1000339
* CVE-2015-7940
* CVE-2013-1624
* CVE-2018-5382
* CVE-2016-1000346
* CVE-2015-6644

hibernate-validator-6.0.18.Final.jar

* CVE-2020-10693

jackson-databind-2.10.2.jar

* CVE-2020-25649
* CVE-2020-36518
* CVE-2021-46877
* CVE-2022-42003
* CVE-2022-42004
* CVE-2023-35116

log4j-api-2.12.1.jar

* CVE-2020-9488

logback-core-1.2.3.jar

* CVE-2023-6378
* CVE-2021-42550

snakeyaml-1.25.jar

* CVE-2022-1471
* CVE-2017-18640
* CVE-2017-18640
* CVE-2022-38749
* CVE-2022-38751
* CVE-2022-38752
* CVE-2022-41854
* CVE-2022-38750

spring-boot-2.2.4.RELEASE.jar

* CVE-2023-20873
* CVE-2022-27772
* CVE-2023-20883

spring-boot-starter-web-2.2.4.RELEASE.jar

* CVE-2023-20873
* CVE-2022-27772
* CVE-2023-20883

spring-core-5.2.3.RELEASE.jar

* CVE-2022-22965
* CVE-2021-22118
* CVE-2020-5421
* CVE-2022-22950
* CVE-2022-22971
* CVE-2023-20861
* CVE-2023-20863
* CVE-2022-22968
* CVE-2022-22970
* CVE-2021-22060
* CVE-2021-22096

spring-web-5.2.3.RELEASE.jar

* CVE-2016-1000027
* CVE-2022-22965
* CVE-2024-22243
* CVE-2024-22262
* CVE-2021-22118
* CVE-2020-5421
* CVE-2022-22950
* CVE-2022-22971
* CVE-2023-20861
* CVE-2023-20863
* CVE-2022-22968
* CVE-2022-22970
* CVE-2021-22060
* CVE-2021-22096

spring-webmvc-5.2.3.RELEASE.jar

* CVE-2022-22965
* CVE-2021-22118
* CVE-2020-5421
* CVE-2022-22950
* CVE-2022-22971
* CVE-2023-20861
* CVE-2023-20863
* CVE-2022-22968
* CVE-2022-22970
* CVE-2021-22060
* CVE-2021-22096

tomcat-embed-core-9.0.30.jar

* CVE-2020-1938
* CVE-2020-11996
* CVE-2020-13934
* CVE-2020-13935
* CVE-2020-17527
* CVE-2021-25122
* CVE-2021-41079
* CVE-2022-29885
* CVE-2022-42252
* CVE-2023-44487
* CVE-2023-46589
* CVE-2020-9484
* CVE-2021-25329
* CVE-2021-25329
* CVE-2022-34305
* CVE-2023-41080
* CVE-2021-24122
* CVE-2021-33037
* CVE-2023-42795
* CVE-2023-45648
* CVE-2024-21733
* CVE-2019-17569
* CVE-2020-1935
* CVE-2020-13943
* CVE-2023-28708
* CVE-2021-43980

tomcat-embed-websocket-9.0.30.jar

* CVE-2020-1938
* CVE-2020-8022
* CVE-2020-11996
* CVE-2020-13934
* CVE-2020-13935
* CVE-2020-17527
* CVE-2021-25122
* CVE-2021-41079
* CVE-2022-29885
* CVE-2022-42252
* CVE-2023-44487
* CVE-2023-46589
* CVE-2020-9484
* CVE-2021-25329
* CVE-2021-30640
* CVE-2022-34305
* CVE-2023-41080
* CVE-2021-24122
* CVE-2021-33037
* CVE-2023-42795
* CVE-2023-45648
* CVE-2024-21733
* CVE-2019-17569
* CVE-2020-1935
* CVE-2020-13943
* CVE-2023-28708
* CVE-2021-43980

**All security vulnerabilities need to be updated to resolve issue and if update doesn’t resolve use NVD or CVE to search for the resolutions for the vulnerabilities.**

**5. Mitigation Plan**

Interpret the results from the manual review and static testing report. Then identify the steps to mitigate the identified security vulnerabilities for Artemis Financial’s software application.

* Encryption of sensitive data
* Two- factor authentication
* Secure data transmission
* Patch security flaws